Cressida Technology Ltd
1 Lammas Gate, 84a Meadrow
Godalming, Surrey
GU7 3HT, UK

Tel: +44 01483 239300
Fax: +44 01483 239383
Email: info@cressida.info
Website: www.cressida.info

Patch Management Best Practices

Whitepaper by:
Chris Roberge, MCSE; CCNA
Product Manager

Table of Contents

Executive Summary

The Problem

Speed, accuracy, and security in sending, receiving and storing information have become key to success in business today. When information systems fail, or become compromised due to a security breach, the loss in time, money, and reputation can be disastrous.

Despite this simple fact, many organizations today do not have an effective maintenance plan in place to protect the assets they value so dearly: information and the systems that protect it.

The Solution

The solution to this problem is an effective maintenance plan for your IT infrastructure. That maintenance plan must include an effective patch management procedure. This document is intended to help you develop your own patch management process by following a series of best practices developed and proven in the field. While each environment's best practices will be slightly different, it is still possible to define a general framework around which you can develop your own best practices.

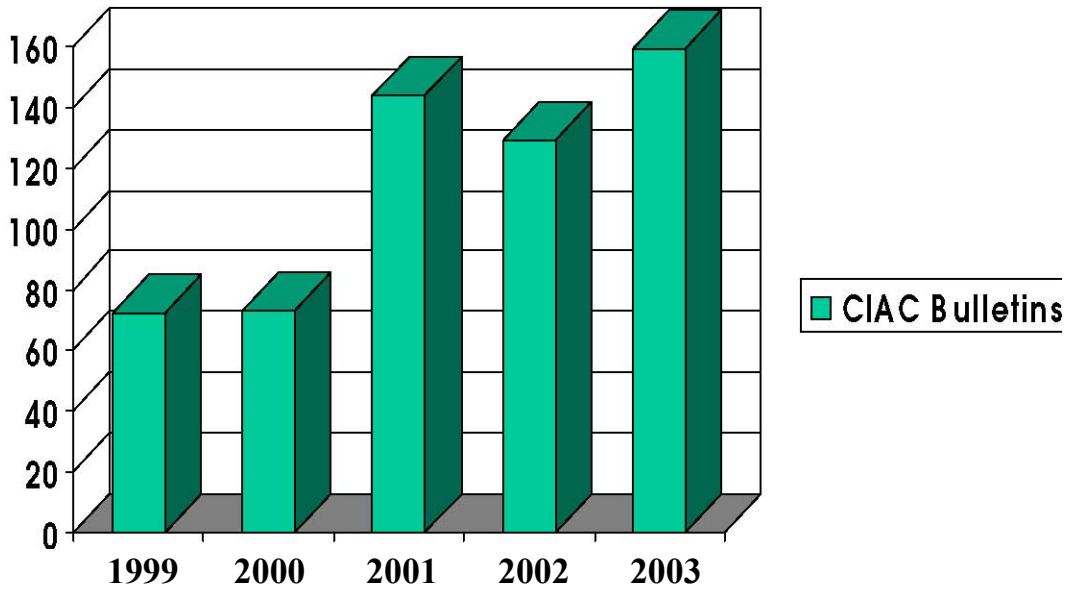
The Challenges of Patch Management

In 2001 System Administrators were already increasingly busy with the day-to-day tasks of running a network. The last thing they needed was yet another job to do. Then along came Code Red and Nimda… and patch management became a new server-room buzzword. After Microsoft's sorely needed "Trustworthy Computing" initiative in the fall of 2001, the patch flood began, and has since escalated to a torrent of regularly released patches from Redmond.

Of course, Microsoft™ is not the only vendor to require patching. Microsoft has the most widely deployed desktop operating system; however, most enterprises have a multiplatform server environment. The need to apply patches consistently and quickly across UNIX®, Linux® and other platforms has also become apparent. There is also a growing requirement for patch management coverage of database management systems and applications, as well. Today, patching has become a process that affects all platforms and applications as more and more security vulnerabilities are being discovered and exploited by more and more sophisticated hackers.

Figure 1 illustrates the number of vulnerabilities reported by the CIAC (Computer Incident Advisories Capabilities) over the last few years and demonstrates the steady increase in the total number of vulnerabilities exposed annually.

(Note Source: CIAC. The CIAC is the division of the US Department of Energy that provides third-party advisories, bulletins and ratings upon discovery of system vulnerabilities. The graph shows the number of Bulletins and Advisories released by the CIAC between 1999 and 2003. Note that the years run from October of the previous year through September of the labeled year.)



The response from the hacking community to the increase in vulnerability identifications has been to step-up their efforts to write code to exploit these vulnerabilities as quickly as possible. In the case of the famous SQL Slammer worm (W32.SQLExp.Worm), the

internet community had six months between the time when the vulnerability was identified (and a patch released) and when the worm was actually released. In the case of Nimda, the lead time was nearly a year. More recently, however, the MS Blaster worm (W32.Blaster.Worm) enjoyed only about a month between discovery and exploit.

This sense of urgency means patches are often released to fix a problem as quickly as possible. There is often no time for vendors to fully test a patch for compatibility with all configurations. This introduces an element of risk to the process of patch deployment. There is no true way to determine the effects of installing a patch in your environment, short of actually installing the patch in your environment.

One apparent solution to this problem is for IT professionals to constantly monitor vendor's websites looking for the latest security patches, to download them and to apply them to the pertinent machines before vulnerabilities can be exploited. (Most vendors offer notification services which can email users upon release of patches that may pertain to the user's environment.) That still leaves a manual download process, a needed determination as to which machines are affected; testing to verify compatibility, and then a process to install those needed patches onto the appropriate systems. Unfortunately, appropriate machines often number in the thousands. Therefore, a manual patching process is impractical.
With so much work involved in patch management, some companies accept the risk of not patching their systems and rely instead on strong perimeter security. Of those who do patch, some patch only their internet-facing systems, such as websites and email servers. Unfortunately, these solutions do not always help. For one thing, relying solely on perimeter security (firewalls, proxy servers, etc.) assumes your perimeter security is flawless, which is not always the case, and viruses are often written specifically to circumvent perimeter security (or sneak through) as in the case of worms and viruses that are spread as either email attachments or embedded within web pages.

The Solution - Patch Management

The solution to this growing problem is to develop a series of best practices. Although the exact procedures followed in each environment will differ slightly, is it possible to define best practices as a series of guidelines that can be customized to your environment. Once you have decided on your best practices, automate those practices through the use of patch management software.

But first - Executive Buy-in

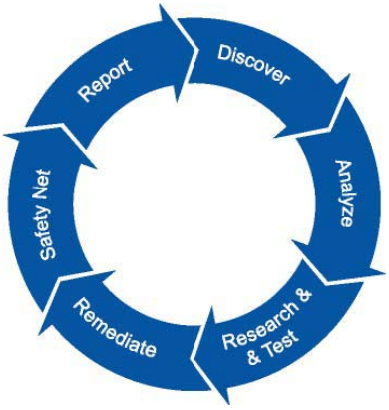
Sometimes the greatest hurdle to overcome is not a technical one. It is crucial, when undertaking any new project, to have the support of senior management. Making senior managers aware of security risks and the need for patches is important for successfully implementing a patch management program and ensuring that appropriate resources are available. Perhaps a quick review of the opening sections of this or any other whitepaper on patch management will help convince them that the need is real and based on financial risk. If not, browse www.ciac.org or www.sans.org and you can usually find all the alarming statistics you'll need to justify an investment in patch management.

Patch Management is not an event, it's a process

Many companies see patch management as something that is event-driven, which is to say, something done in response to an outbreak of some kind. For example, during the SQL Slammer outbreak in early 2003, companies scrambled to install patches across their SQL Server farms. Unfortunately, Slammer took all of nine minutes to spread worldwide. (Not much time to deploy a patch, let alone research and test one.) This event-based patching philosophy is akin to fixing the barn door after the Trojan horse has come home. The time to patch a given vulnerability is *before* it is exploited. *After* it has been exploited is too late and, in many situations, may necessitate a full rebuild of the affected systems.

Therefore, it's important to look at Patch Management as a process, ideally a closed-loop process. A closed-loop-process an automatic control system in which feedback acts to maintain output at a desired level. This means essentially that patch management should be automated to the point where it can maintain your desired patch levels with as little human intervention as possible. Patch management, as an automated series of best practices, has to be repeated regularly on your network to ensure protection from exposed vulnerabilities. Patch management requires the regular re-discovery of any systems that may potentially be affected, scanning of those systems for known vulnerabilities, download of patches and patch definition databases, deployment of patches to the systems that need them, and verification of installation.

Defining the Best Practices

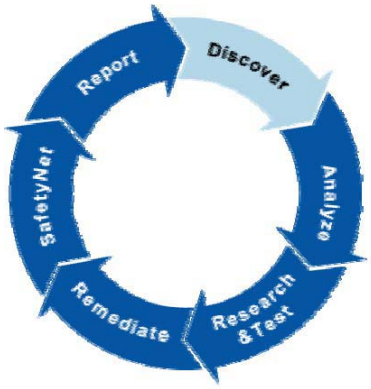


Ecora has developed a six-step method to Patch Management. These six steps, discussed as a closed-loop solution, define an effective framework for patch deployment whether you are bringing an un-patched environment up to a baseline level or deploying a patch as part of an emergency response plan. The six steps in the Ecora method are:

. • **Discover** – The discovery phase involves locating assets (workstations and servers) on your network and categorizing them.
. • **Analyze** – Through the analysis process, current patch levels must be determined and a minimum baseline policy should be defined.
. • **Research and Test** – In this phase, missing service packs and patches must be researched and understood. A risk analysis must be done for missing patches.
. • **Remediate** – To "remedy" the vulnerabilities found by bringing systems up to date. This is best accomplished via policy-based solutions.
. • **Safety Net** – The safety net, although not always a necessary step, describes the ability to roll back a patch should the need arise.
. • **Report** – Reporting conducts a change review and verifies successful deployment of patches. Reporting should also include enough review, analysis, and adjustment to close the loop and begin the cycle again automatically.

The following sections will look at this process in greater detail.

Step One: Discover



The first step in Patch Management is to define your starting point. You must develop a clear and accurate picture of what is needed on your network to get your patch situation under control. The first step is to identify and categorize your assets: taking a full inventory of all workstations and servers on your network. Typical IT environments often include dozens to hundreds of servers and hundreds to thousands of workstations. Locating and documenting each of those systems manually represent an enormous undertaking. Therefore, many patch management products include some method to scan a network and locate hosts. There should be multiple discovery methods available, from Active Directory computer account location to the IP address and subnet scan, to ensure that the discovery phase is as complete as possible.

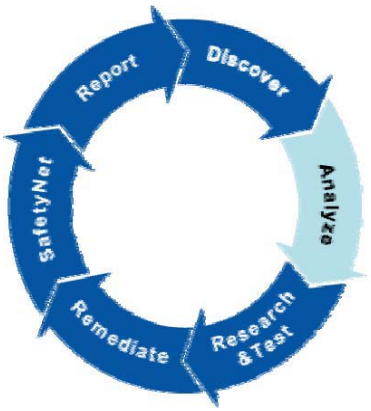
Once your assets are identified, they need to be categorized based on exposure and risk. By categorizing assets, you develop a picture of which machines require rapid patch management (within hours or days) and which require standard management (weeks.) Categorizing your assets is almost always a manual process. It's difficult to automate a process that essentially identifies "important machines" and "less-important machines." One consideration when categorizing machines is the information that machine protects. Other issues to consider are public visibility (as in the case of a website) and sensitivity (customer credit card numbers). The most important question to ask is "what will be the impact on the company if this machine is down or compromised?"

(Note: Risk Analysis should be an integral part of the Patch Management process. Please see the "Useful Links" section at the end of this paper for more information.)

Most patch management applications support the concept of system grouping. Within the application, there is the ability to create logical groups of computers based on risk, configuration, department, physical location, or whatever criteria the administrator requires. There also should be a capability to group systems by the roles they perform (for example SQL Servers, IIS boxes, Domain Controllers and File Servers.) These groups may be broken into sub-groups of high-risk and low-risk machines and systems must be able to belong to multiple groups to be truly useful for deployment.

The network should be periodically "re-discovered" via an automated mechanism to capture information about any additional systems that are brought online or removed from the network. How often this rescan takes place depends heavily on how often systems come and go and are rebuilt on your network. Rescans should happen more frequently on networks that change often, and happen less frequently on more stable networks. What is most important is that there is a process in place to capture information regarding changes on the network.

Step Two: Analyze



The next step is the analysis phase, in which current patch levels are assessed. Done manually, this requires researching every system's configuration and current levels, which is not feasible with most staffing levels. Patch management applications are designed to scan the systems they discover for installed and missing patches. The accuracy of this step is critical. Worst case scenarios are false-positives; reporting a patch as present when in fact it is not. This may result in the patch never being applied. The less-critical counter to this is a false-negative; reporting a needed patch is not present when in fact it is. This will usually result in the re-application of the patch, with little harm done.

This patch analysis is based on several different information points. Typically, the operating system needs to be determined for a given device, as well as which applications installed on the machine. Based on that information, most tools will consult a "master list" of patches that are available for a given OS and application and determine which of these patches are installed and which are not. This "Master List" is analogous to antivirus software virus definition files and should be downloaded regularly from vendor websites. Most patch management products can download these files automatically, and are able to determine which patches conflict with other patches, which ones supersede others, and take into account service packs and other types of collective patch rollups.

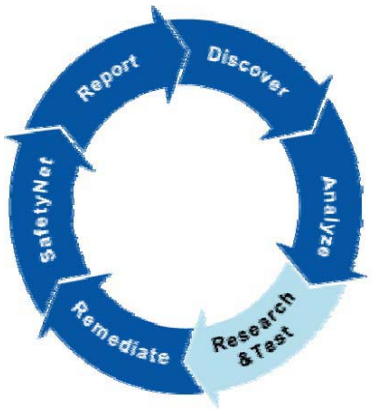
Based on this information, patch management products display a report of patches that are installed and missing on each system. Many applications allow you to view the data in different ways, depending on what specific piece of information you're seeking. For example, Ecora's Patch Manager features 3-D Patch Views™; three distinct views of your systems patch levels. The Hosts view allows you to view by host name and look for specific machines to determine their current patch level. The Products view shows the complete list of products supported by Patch Manager, which machines are running those apps, as well as the current patch levels. The Patches view allows you select a specific patch and see which machines have or do not have a patch.

This last view can be helpful if you are looking for all instances of a specific vulnerability across your network. You should perform a network analysis within 24 to 48 hours of the release of a new patch to determine your network's exposure to the vulnerability. Based on this information, as well as severity and risk information, you will have a better understanding of how vital a patch is to your network security.

Initially, your first steps on an unpatched network will be the analysis, to verify on which machines a particular service pack or patch is installed, and on which machines it is missing. Any machines that fall below your minimum baseline have to be brought into

compliance. (Note: Patch Manager features a component called Policy Manager which automates definition, analysis, and deployment of baseline policies to multiple systems.)

Step Three: Research and Test



Let's consider the results of the first two steps: You should now have a clear picture of your current patch levels. Your current patch levels will fall into one of the following three categories: 1) Fully-patched, in which all of your systems are completely up-to-date; 2) Totally unpatched (Windows 2000, no service packs) or 3) Somewhere-in-the-middle. For those with networks in the first category: do not sit back and relax; there will always be new patches to deploy. For those in the second and third categories, read on. Patch Level Minimum Baselines

An important concept is the minimum patch level you require on your network. This minimum patch baseline will be unique to each network and can only be determined by a thorough understanding of the analysis, research, and test phases. Consider a series of unpatched machines, then ask the following question: what patch level do I want to achieve? Do I want every possible patch deployed, regardless of severity? Am I happy having Windows 2000 machines at Service Pack 4? If you feel comfortable with Windows 2000 Service Pack 4, you may choose to define that as your minimum baseline. If you know several machines on your network are susceptible to a given vulnerability, the required patch for that vulnerability should also be part of your baseline.

Some administrators are happy with service packs. Service packs are essentially rollup packages of bug-fixes, security-fixes, and feature enhancements that are released every six to twelve months. They are usually beta-tested in production environments and fully tested by the vendors. Because of the extensive testing, they usually represent the most stable and reliable operating system or application updates you can install. For this reason, many administrators will not install patches until they are released as part of a service pack (with the possible exception of high-severity patches for vulnerabilities with active exploits in the wild.) For these administrators, a Windows 2000 system with the current service pack installed may represent a well-patched system. For other environments, a well-patched system is one in which not only the latest service packs are applied, but all post-service pack patches are in place as well. Whatever the case may be the latest service packs will generally represent the best place to start.

Research

Before you begin the process of deploying any service packs or patches to your network, it is STRONGLY recommended that you research what you are about to deploy. Occasionally, the application of a patch, or even service pack, can have an unexpected negative impact on a machine; therefore it is necessary to understand what you are deploying to your network.

To this end, Ecora and other vendors provide independent engineering notes from patch testing. Ecora tests patches in-house and notes information regarding incompatibilities. Review resources such as the CIAC website, where vulnerabilities are reviewed and detailed. Vendors publish articles describing vulnerabilities and include release notes and/or a read-me files describing installation options and precautions. Vendor testing should never replace your own, however. Every environment is different and third-party or custom software makes interactions unique and unpredictable. Based on information you collect, you should determine the following for each patch you deploy:

1) What is the nature of the vulnerability? What application or OS component is affected by it? How easy is it to exploit the vulnerability? 2) What is the severity of the vulnerability? If the vulnerability is exploited, how much damage could be caused? Vulnerabilities are typically rated as low, medium, high or critical, critical being the highest level of potential damage should the vulnerability go un-patched. 3) What is your level of exposure to the vulnerability? How many (if any) machines on your network are affected?

Use the above information to guide your deployment of patches. Conduct a risk analysis. For example, if you find a high occurrence of missing patches for severe vulnerabilities, you may wish to address those systems first. Is this a severe vulnerability on your mission-critical application servers, or is it a low-severity across internal workstations? Based on that determination, you can begin to address the issues of testing the new patches for deployment.

A Few Precautions

It should also be noted that, in the case of major system upgrades (and some small ones, too), reasonable precautions should be taken before making any change. This includes reading release notes and any deployment guide. There may be recommendation to back up critical data or the entire system before deployment, so read carefully.

Test

The reason for testing patches prior to deployment may be obvious, but should be stated clearly: patches sometimes break operating systems. It's a fact of patch management. Even in the case of a fully tested service pack, there is always a chance that it will conflict with some as-yet-undiscovered quirk in a small number of environments and, when that conflict occurs, servers come down. Therefore, the importance of testing in your own environment, on your own machines, can't be stressed enough.

The testing phase of deployment includes applying patches to a test environment prior to deploying them to a production system. The nature of a patch is that it has been written

quickly to address a critical issue. Therefore, there is not always time to thoroughly test a patch prior to release. This is not to imply that patches are untested; but the testing isn't nearly as extensive as in the case of a service pack, which goes through beta testing and review prior to release. Of course, service packs should not be immune to the testing phase. Although they are tested thoroughly by their vendors, no vendor can test every update in every possible environment, so no patch or service pack should ever be deployed without being tested in your own test environment first.

So how do you test a patch or service pack? Deploy it to a test machine configured like the production system(s) that need the patch. Ecora highly recommends that you develop a test environment and use that environment to test patch deployment before deploying to production. Large corporations often have a lab which contains enough systems to create an environment that mimics the actual corporate network, complete with Domain Controllers, servers and workstations. Smaller companies often settle for a test environment that consists on one or two machines configured exactly like their production machines or virtual machines loaded and reloaded based on what needs testing. At the bare minimum, if a test environment is not available, patches should be deployed to, and tested on, low-priority production system first.

Whatever your test environment, create a logical group within your patch management software to hold the machines within it, and deploy patches that need testing to that group first. Then observe and record the results. Is the system still functioning? Are the applications and services on it still functioning? Do the results of the install coincide with the expected results (application extensions are updated, registry keys are changed?) If no negative impact is determined, the patch can be deemed safe. If a problem occurs, go back to the research phase. Check websites such as www.ntbugtraq.com to determine if anyone else is experiencing problems like yours and if there is a workaround. Determine the root-cause of the problem and decide if deploying the patch is still worthwhile

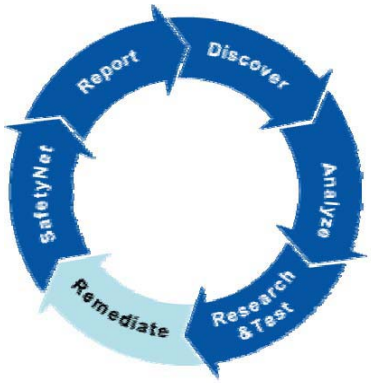
Testing using Image-Based Systems

Let's take a moment to discuss image-based system deployment and how it relates to your patch management process. Image-based system deployment concerns using imaging (or cloning) software to create a "master" image of a computer hard drive. This image can then be compressed and stored on a server or CD-ROM. The image is created by literally pulling the data off a manually installed and configured system block-by-block and storing that data in a compressed file format. The image is that of a fully configured operating system, including applications, settings, and, in many cases, patches and service packs. The image can then be copied to a "bare-metal" system that, once rebooted, is an exact replica (or clone) of the original system.

This can be advantageous when testing patches and service pack. Most organizations who use the cloning process have several images stored that represent individual workstation or server configurations on their networks. Therefore, when testing a patch, one of these

images could be copied onto a bare-metal system, new patches could be deployed to it, the stability verified, the patches approved for production, the master image updated, and the patches rolled out to production.

Step 4 – Remediate



Remediation is the act or process of remedying; concerned with the correction of a faulty situation. Remediation in the context of software means to correct, update, patch, or rollback to bring a system into compliance, therefore this phase involves patch deployment, installation, and un-installation (if necessary) in a controlled manner.

The remediation phase is actual patch download and deployment. Remediation occurs during your initial pass at bringing your network up to the minimum baseline, every time a new computer is brought online, and every time a new patch is released that applies to any systems on your network. This is where the automation of patch deployment is most critical.

Because downloading patches is critical to all phases of deployment, most patch management applications can be configured to regularly contact the vendor websites and download the most current patch-definition database and any new patches available.

Incremental Rollout

It is strongly recommended that patches be deployed incrementally. Rather than blanketing a patch out to thousands of machines at once, follow the above testing recommendations, analyze the results, and then deploy to small groups of machines. This is a good way to identify incompatibilities without the potential of wreaking havoc on the production network due to a bad patch. Once the patch is deemed ready for production deployment, start with just a portion of your environment. This portion could be a single subnet, or perhaps a department. Following successful deployment to that subset, deploy to another subset, then another. Depending on the size of your environment, go as quickly or as slowly as you are comfortable. One advantage of this method is that, should problems arise, patches can be rolled back from subsets.

(Note: In the testing phase of deployment you have technically already begun the remediation process. In that case, a given vulnerability has been removed from your test environment. During production remediation the same vulnerability will be removed across the enterprise.)

Scheduling Reboots

Another consideration for patch deployment is the reboot that vendors sometimes require following the installation of a patch or series of patches. In many environments, it is not feasible to have a production server down for any length of time during peak hours. It is important to be able to schedule the installation of patches, especially those that require

reboots, for off-peak hours or weekends, or to at least be able to defer the reboot of the computer until a more convenient time.

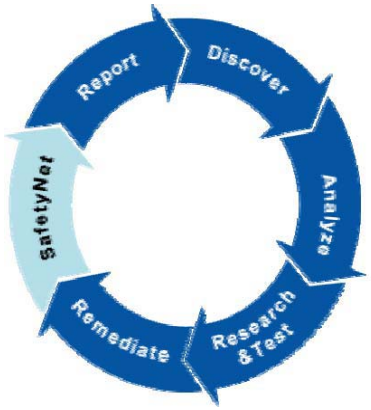
Policy-based Remediation

Policy-based remediation is essential to an effective patch management strategy. Policy-based remediation promotes deployment by patch-level baselines and rule-based remediation. For example, a policy allows you to create a rule like "All Windows 2000 Service Pack 3 machines, with MDAC 2.6 installed, must have xyz patch installed." Once the rules and conditions are set, policy-based solutions can enforce this policy across the enterprise or selected relevant groups. Policy-based remediation of multiple patches across multiple machines should be considered an extension of the concept of baselines. It is essential that your patch management application allow you to create policies, analyze compliance to policies, and deploy based on non-compliance.

A Few Precautions

It should also be noted that in the case of major system upgrades (and some small ones, too) that reasonable precautions should be taken before making any change. This includes reading release notes and any deployment guide. There may be recommendation to back up critical data, or perhaps the entire system before deployment, so read carefully.

Step Five – The Safety Net



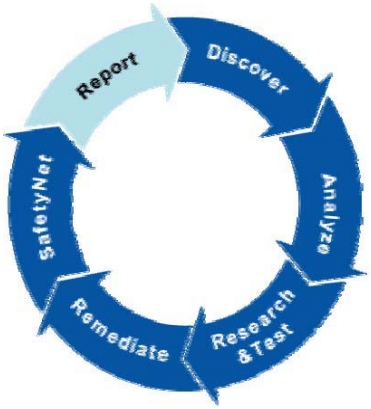
Step five should not, hopefully, require constant attention, but may become necessary in the event that an applied patch causes problems on your network. The safety net is employed when a patch requires rollback.

Rollback is essentially the ability to uninstall a patch and restore the system to its prior state. In the event that a patch does cause problems, the ability to uninstall the patch is highly desirable. Patch Manager allows patches to be rolled back individually or removed from policies, depending upon the nature of the patch (whether or not the vendor supports rollback). It is important to select a patch management solution that allows for the convenient roll-back of any patch that supports it.

Rollback can also be important in the event that a patch was deployed without going through the proper authorization. Many companies employ a change management policy. This essentially describes the processes and procedures that must be followed when any change needs to occur. In many cases, the deploying of a patch or service pack is considered a configuration change which requires approval or authorization.

Since roll-back support is not universal to all patches from all vendors, it would be wise to include a procedure or process for documenting configurations and tracking changes in your best practices. Before and after snapshots of system settings and registry keys allow manual restoration of a system if necessary.

Step Six – Reporting



Reporting is the final step in the Patch Management process. You must be able to confirm the successful deployment of patches and verify that there is no negative impact. Reporting should expose situations that require an immediate return to the analysis phase, such as a failure in deployment. Reporting also allows an opportunity to review patch management process and look for areas of improvement, as well as providing valuable statistical information regarding patching activity. In environments where internal or external audits (often to meet industry or federal regulations) are required, documentation of changes is crucial to proving compliance.

Return to Step One - Close the Loop

These six steps bring us to the end of our closed-loop process of patch management, which is back to the beginning. It is not possible to understate the need to repeat each of these steps as often as possible and automation and scheduling can make the loop reasonably self-maintaining. For some networks, this will be daily, for some, weekly, and others, monthly. The preceding six steps should be added to the regular maintenance plans for your enterprise, along with the defragmenting of your drives and updating of antivirus software. Only by automating proactive approaches can you hope to stay ahead of patch management.

Additional Considerations

Choosing the right patch management product

There are several considerations for choosing the right patch management product for your environment. (You *will* have to choose one. If nothing else is clear by now, what should be is that these processes can only be effective if automated.) Some important considerations include:

- Platform Support: Are all of the operating systems present in your environment supported? One of the most popular products for deploying patches to Microsoft networks is their Software Update Service. Unfortunately, UNIX environments benefit little from this product.
- Application Support: Are all of the applications present in your environment supported? At least the most vulnerable or most critical to the business?
- Usability: How is the learning curve? When choosing any product, you must strike a balance between ease-of-use and functionality. It will do no good to buy the latest and greatest product if the interface is confusing and time-consuming.
- Features: Choose a product that allows for as much scheduling and automation as possible. Some of the things your product should include:
    - Scheduling and alerting
    - Discovery of servers and workstations
    - Resource grouping
    - Quick, accurate, and flexible analysis
    - Policy-based analysis and remediation
    - Automated roll-back support
    - Reporting
- Agent-based vs. agentless: One hot debate is agent-based vs. agentless deployment. Benefits of agent-based solutions are that they generally provide more functionality, consume less network bandwidth, and support mobile users via local client scanning technology. Drawbacks include the time, money, and

resources required to deploy the agents and any destabilizing effects they may have on the clients and workstations.

- Cost: In many cases this may be the greatest concern. Tool range from "free" to "how much have you got?" and generally provide more features and functionality as you go. Remember to consider total cost of ownership; figure in cost of implementation, training / learning curve, customer service or maintenance, etc.

Security Policies Best practices exist for patch management that go beyond the scope of any patch management software. Security Policies are written documents that describe expectations regarding all aspects of security in networked environments. These policies can cover everything from Internet usage to password policies and should ideally describe how users should handle email attachments, unsolicited email, unknown web sites, and other common conduits for viruses and worms. A good security policy will contain provisions for patch deployment, describing how and when new patches should be applied to the enterprise, and acceptable "discovery-to-patch" timeframe. A security policy will also discuss how the policy is enforced and audited, as well as how violations are handled.

Change Management Plans Often undocumented or uncoordinated changes can have a serious negative impact on a system or network; therefore it is important to put in place controls that will prevent such changes from happening arbitrarily. A change management plan is a written procedure designed to require an approval process for a change to take place, as well as the procedures for carrying out the change. By following the written plans, it is less likely unexpected changes will occur and, if planned changes wreak havoc, they are documented, therefore easier to roll back.

Emergency Response Plans A well-documented emergency response plan is a document that describes what to do in the event of an emergency, from a single computer security compromise to a full-scale natural disaster. These plans typically include emergency telephone contact numbers, evacuation plans, and business-continuity plans (BCP) in the event of a total-asset loss, such as destruction of a building.

Conclusion

Effective patch management has become a necessity in today's information technology environments. Reasons for this necessity are:

1. The ongoing discovery of vulnerabilities in existing operating systems and applications,
2. The continuing threat of hackers developing applications that exploit those vulnerabilities, and
3. Vendors' requirement to patch vulnerabilities via the release of patches. These points illustrate the need to constantly apply patches to the computing environment. Such a large task is best accomplished following a series of repeatable, automated best practices.

Therefore, it's important to look at patch management as a closed-loop process. It is a series of best practices that have to be repeated regularly on your network to ensure protection from exposed vulnerabilities. Patch management requires the regular rediscovery of systems that may potentially be affected, scanning of those systems for vulnerabilities, downloading patches and patch definition databases, and deploying patches to systems that need them. To recap the six-steps:

- **Discover** – The discovery phase involves locating assets (workstations and servers) on your network and categorizing them.
- **Analyze** – Through the analysis process, current patch levels must be determined and a minimum baseline policy should be defined.
- **Research and Test** – In this phase, missing service packs and patches must be researched and understood. A risk analysis must be done for missing patches.
- **Remediate** – To "remedy" the vulnerabilities found by bringing systems up to date. This is best accomplished via policy-based solutions.
- **Safety Net** – The safety net, although not always a necessary step, describes the ability to roll back a patch should the need arise.
- **Report** – Reporting conducts a change review and verifies successful deployment of patches. Reporting should also include enough review, analysis, and adjustment to close the loop and begin the cycle again automatically.

By following these six steps and repeating them regularly, the process of bringing your network into patch compliance quick, effective, and accurate.

Useful Links

For more information on Risk Analysis, visit The Society for Risk Analysis at:
http://www.sra.org/

For more information on Change Management, visit the Change Management Resource Library at:
http://www.change-management.org/

For information regarding general security best practices, visit
The Rainbow Series Library at:
http://www.radium.ncsc.mil/tpep/library/rainbow/
and Microsoft's Security Center at:
www.microsoft.com/security

We hope you have found this white paper useful.
Please email dpratt@ecora.com with any comments or suggestions.